Published in: Proc. of Net.ObjectDays 2001, Erfurt, Sept. 2001, pp. 254-259

Performanz von XML-Datenbanksystemen

Rita Schindler, Günther Specht, Andre Schmatloch

Fakultät für Informatik und Automatisierung
TU Ilmenau
Postfach 100565, D-98684 Ilmenau
{rita.schindler, gspecht}@prakinf.tu-ilmenau.de

Abstract. Die heute rasch zunehmende Menge an XML-Daten verlangt immer mehr nach einer effizienten Speicherung und Indizierung in Datenbanksystemen. In diesem Papier stellen wir erste Performanzmessungen bezüglich Einspoolen, Suchen, Ändern und Wiederzusammensetzen komplexer XML-Dokumente für vier völlig unterschiedliche XML-Datenbanksysteme vor: Tamino, eXcelon, Poet und Oracle, letztere in beiden Varianten: nested Tables und Varrays. Dabei ergeben sich für ein zweidimensional skalierbares „Real-World"-Beispiel (Lieferanten und Produkte), erstaunlicherweise gänzlich unterschiedliche Ergebnisse und Eignungen.

1. Einleitung: XML-Datenbanken

XML entwickelt sich zunehmend zum Standard als Austauschformat für Daten aber auch als Erfassungs- und Ablageformat großer Mengen semistrukturierter Daten. Vom E-Business-Bereich bis zu digitalen Bibliotheken wird heute bereits XML als grundlegende Datenerfassungs- und Speicherstruktur verwendet. Die meisten XML-Dokumente werden aber noch in Ascii-Dateien gespeichert und verwaltet. Die zunehmend große Datenmenge macht jedoch eine Speicherung und Verwaltung in Datenbanken immer notwendiger. Insbesondere geht damit das Problem einer effizienten Suche in den XML-Datenbeständen einher.

Erste naive Ansätze speicherten dazu DTD und XML-Dokumente als Ganzes in CLOBS (character long objects, also Text-BLOBS) in relationalen Datenbanken. Danach folgten Ansätze, XML-Daten in zwei Relationen (eine für alle ELEMENTS und eine für alle ATTRIBUTS) aufzuspalten. Erst neuere Ansätze erlauben die strukturierte Speicherung und Suche von XML-Daten in Datenbanken. Zur Zeit werden, aufgrund großer Markthoffnungen fast alle relationalen, objektorientierten und objekt-relationalen Datenbanksysteme mit entsprechenden Extendern versehen und demnächst angeboten. Dass dabei nicht alle gleich gut zur effizienten Erfassung, zur effizienten Suche und zum effizienten Wiederauslesen des gesamten XML-Dokuments geeignet sind, liegt auf der Hand. Wir haben daher vier ausgewählte XML-fähige Datenbanksysteme (vielleicht die z.Zt. wichtigsten Systeme) genauer untersucht und mittels Performanzmessungen in einem praxisnahen eigenen Benchmark verglichen. Untersucht wurden die Systeme:

- Oracle 8i in den Speichervarianten:
    1. Nested Tables
    2. Variable lange Arrays
- eXcelon (B2B Portal Server 2.5) von eXcelon Corporation
- Tamino 2.2.1. von der Software AG
- Poet Management Suite 2.5 von Poet

Damit wurde auch gleichzeitig das ganze Spektrum unterschiedlicher zugrundeliegender Datenbanktechnologien abgedeckt. Während Oracle ein objektrelationales Datenbanksystem mit relationalen Wurzeln ist, ist Poet ein rein objektorientiertes Datenbanksystem. Tamino hat als Vorgänger ADABAS C, welches ein hierarchisches Datenbanksystem ist. Da auch XML eine hierarchische Baumstruktur beim Parsen aufbaut, wurde hier versucht diese aufeinander abzubilden. Excelon lässt sich im weiteren Sinne ebenfalls als OO-Datenbanksystem charakterisieren, wobei hier jeder Knoten des Parsebaums durch ein eigenes Objekt repräsentiert wird.

Als Benchmark kommt eine Standardanwendung zum Einsatz: Lieferanten und deren Produkte in jeweils variierbarer Anzahl. Dabei zeigen sich ganz erstaunliche Unterschiede in den Performanzmessungen bei der Speicherung und Suche in den XML-Datenbanken.

Der Rest des Papiers ist wie folgt aufgebaut: Abschnitt 2 bringt Arbeiten Anderer, wobei diese zur Zeit vor allem auf dem Gebiet der Erstellung möglicher Benchmarks liegen, noch nicht auf den Messungen. Abschnitt 3 zeigt unser Test-Szenario, Abschnitt 4 bringt die eigentlichen Performanzmessungen und Abschnitt 5 fasst die wichtigsten Ergebnisse nochmals zusammen.

2. Andere Arbeiten

In der hier vorliegenden Arbeit wurde eine Test-Datenbank mit XML-Daten konzipiert und unter verschiedenen XML-fähigen Datenbanksystemen implementiert. Die anschließenden Messungen beziehen sich auf das Zeitverhalten der Datenbanksysteme bei typischen Such- und Änderungs-Operationen auf den XML-Daten unabhängig von Synchronisationsproblematiken (Einzeltransaktionsbetrieb).
Den Autoren sind keine vergleichbaren und veröffentlichte Messungen bekannt. Allerdings arbeiten mehrere Gruppen an der Entwicklung von entsprechenden, auf die unterschiedlichsten Aspekte ausgerichteten Benchmarks. Von diesen Arbeiten sollen zwei der neuesten kurz vorgestellt werden.
So ist XMach-1 (XML Data Management Benchmark, siehe [2]) ein Benchmark zur Evaluierung von XML-Datenverwaltungssystemen und XML-fähigen relationalen DBMS. Er umfasst eine Datenbank mit verschiedenen Typen von XML-Daten (Text-Dokumente, schemalose Daten und strukturierte Daten) und einen Satz von typischen Update-Operationen und Anfragen auf diesen Daten. Gemessen wird die Durchsatz-Performance für die zu testenden Datenbanksysteme im Mehrbenutzer-Betrieb, basierend auf Web-Applikationen.

Der Xmark Benchmark – ausführlich beschrieben in [5] – konzentriert sich auf das Austesten des Anfrage-Prozessors von XML-Datenverwaltungssystemen. Im Rahmen des Benchmark Projekts werden eine skalierbare Dokumenten-Datenbank (eines Auktions-Hauses), ladbare Spezifikationen und eine kompakte Menge von 20 Anfragen mit XQuery zur Verfügung gestellt. Die Anfragen sind ausgerichtet auf die grundlegenden Aspekte des XML-Prozessing in Datenverwaltungssystemen wie Pfadausdrücke, Anfragen mit NULL-Werten, Volltext-Suche, Konstruktion von komplexen Resultaten und weitere. In der zitierten Arbeit werden auch erste Messergebnisse zum Antwortzeitverhalten in einer experimentellen Umgebung vorgestellt.

3. Die Test-Datenbank

Das Testszenario ist eine skalierbare Lieferantenkartei. Die entsprechenden XML-Dokumente beinhalten Daten zu den Lieferanten mit Angaben zur Anschrift und den angebotenen Produkten. Ein Lieferant kann mehrere Produkte anbieten, aber eine Produktinstanz kann nur von einem Lieferanten geliefert werden. Die zugehörige DTD ist wie folgt definiert:

```
< ! ELEMENT  LIEFERANTENKARTEI  ( LIEFERANT* ) >
< ! ELEMENT  LIEFERANT  ( DATEIID ,  LIEFID ,  STAMMSITZ ,  ( PRODUKTE* ) ) >
< ! ELEMENT  DATEIID  ( #PCDATA ) >
< ! ELEMENT  LIEFID  ( #PCDATA ) >
< ! ELEMENT  STAMMSITZ  ( FIRMENNAME ,  STRASSE ,  PLZ ,  ORT ) >
< ! ELEMENT  FIRMENNAME  ( #PCDATA ) >
< ! ELEMENT  STRASSE  ( #PCDATA ) >
< ! ELEMENT  PLZ  ( #PCDATA )
< ! ELEMENT  ORT  ( #PCDATA )
< ! ELEMENT  PRODUKTE  ( PRODUKTE_ITEM* ) >
< ! ELEMENT  PRODUKTE_ITEM  ( BEZEICHNUNG ,  BESCHREIBUNG ) >
< ! ELEMENT  BEZEICHNUNG  ( #PCDATA ) >
< ! ELEMENT  BESCHREIBUNG  ( #PCDATA ) >
```

Während der Testläufe wurde die Dateigröße kontinuierlich erhöht (Anzahl der Lieferanten von 10 über 50, 100, 200, 500, 1000, 2000 bis 3000 und die Anzahl der lieferbaren Artikel pro Lieferant von 3 über 10 bis 20). Dies entspricht einer Skalierung sowohl in der Länge als einer Skalierung in der inneren Verzweigung. Insgesamt stehen damit 8 * 3 * 5 (Systeme) * 5 (Operationen: Einfügen, Suchen, Ändern, Expotieren, Löschen) * 5 (Wiederholungen) = 2400 Messungen in 60 Messkurven an, die insbesondere zum Ziel haben, die Eignung der einzelnen Datenbanksysteme bezüglich den üblichen Operationen (Einfügen, Suchen, Ändern, Löschen und Wiederzusammenbauen (Exportieren) von XML-Daten) zu untersuchen.

Wie beschrieben, wurden bewusst unterschiedliche Datenbanksysteme, die auf verschiedenen Technologien beruhen, ausgewählt:

1. Oracle8i - relational bzw. objektrelational
2. eXcelon B2B Portal Server 2.5 - objektorientiert
3. Poet Content Management Suite 2.5 - objektorientiert
4. Tamino 2 - nativ XML-Datenbanksystem

Der Zugriff auf die Datenbanken erfolgte mittels Java (bei POET unter Verwendung des POET Content Client).

Es wurde unter folgender Systemkonfiguration gearbeitet:

- Prozessor: Pentium PIII Celeron 450MHz,
- Hauptspeicher: 256 MB
- Betriebssystem: Windows 2000 SP1

Dabei wurde die Java VM mit folgenden Parametern für die Speicherreservierung gestartet:

-Xms150M -Xmx300M

Um Verzerrungseffekte durch die spezielle Art der Speicherfreigabe des Garbage Collectors des Java-Interpreters zu vermeiden, wurde vor jedem Durchlauf (Test) sichergestellt, bis mindestens wieder 100 MB Hauptspeicher zur Verfügung standen. Die Datenbanken waren jeweils lokal installiert, um Einflüsse durch das Netzwerk beim Zugriff auszuschließen. Für jeden Test wurden 5 Durchläufe realisiert, um den Einfluss eventuell doch noch auftretender nicht periodischer Ereignisse des Systems zu verringern. Der Durchschnitt der jeweils ermittelten Zeiten ergibt den Ergebniswert für eine Testsituation.
Für die unterschiedlichen Datenbanksysteme war es nicht möglich, gleiche Umgebungsparameter festzulegen. Deshalb wurden die Tests mit den optimalen Einstellungen für die jeweilige Datenbank durchgeführt (Cachegröße, Suchschlüssel).

Es wurden Ergebniszeiten für folgende Datenbankoperationen ermittelt:

1. XML-Datei importieren
2. XML-Datei exportieren
3. Lieferanten suchen und exportieren, der im ersten Drittel der Lieferantenliste steht
4. Daten des Lieferanten ändern, der sich im zweiten Drittel der Lieferantenliste befindet
5. Löschen der importierten XML-Daten

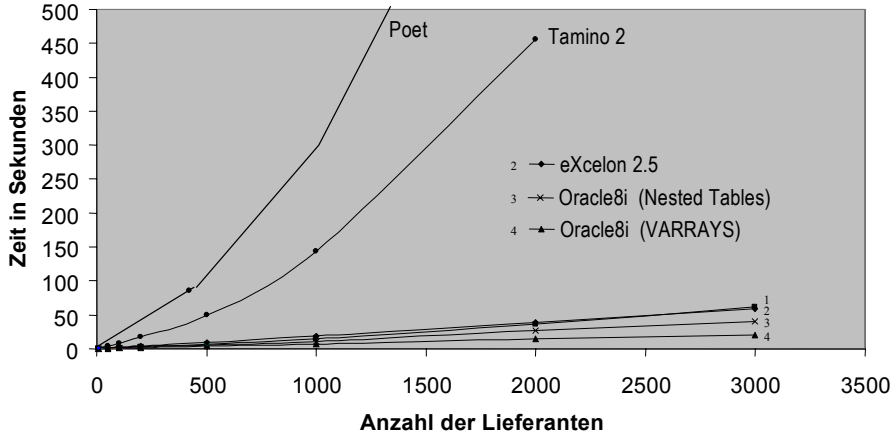
Bei Poet war es aufgrund fehlender APIs nur möglich das Importieren, Exportieren und Löschen zu messen.

4. Ergebnisse

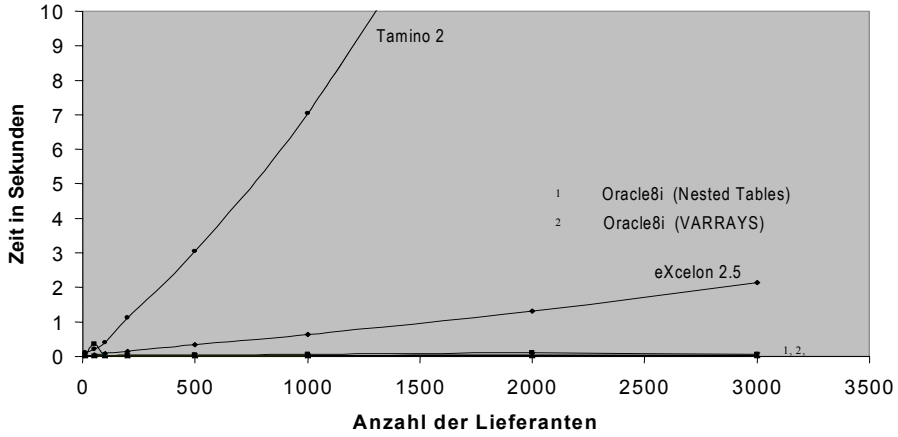
Da alle Einzelmessungen den Rahmen sprengen würden, wird im folgenden die grafische Darstellung der Messergebnisse für das Importieren, Exportieren und Suchen angegeben. Dabei wird in der Darstellung über die Anzahl der Lieferanten variiert. D.h. die hier ausgewählten Kurven skalieren im wesentlichen über die XML-Datenmenge bei konstanter Strukturtiefe. Die XML-Hierarchietiefe hat 6

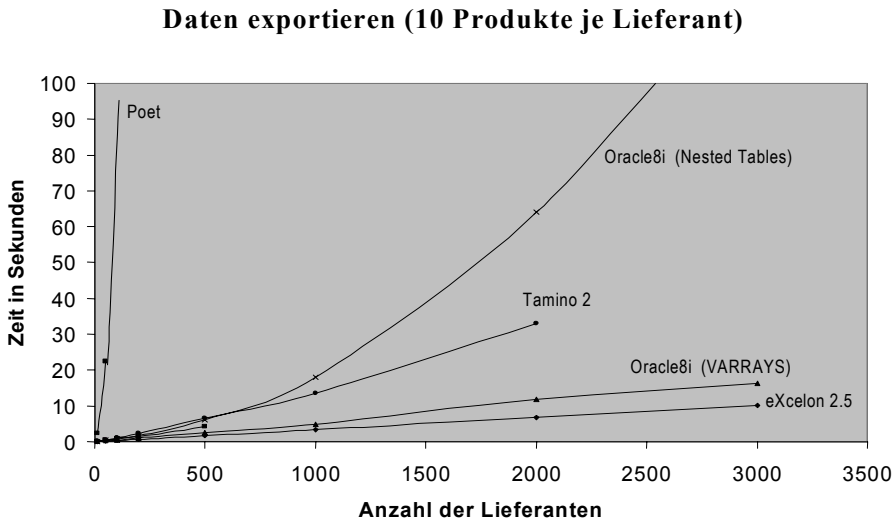
gefüllte Ebenen je Lieferant, bei konstantem inneren Verzweigungsgrad (je Lieferant 10 Produkte), wobei beides typische Zahlen für XML-Dokumente sind. Die Varianz über den inneren Verzweigungsgrad und die nichtgezeigten Kurven anderer Einstellungen zeigen aber alle im wesentlichen das gleiche Verhalten. (Für die ausführlichen Ergebnistabellen siehe [4, S.35-39]).

Daten importieren (10 Produkte je Lieferant)



Daten suchen (10 Produkte je Lieferant)





6. Zusammenfassung

In diesem Papier stellten wir erste Performanzmessungen bezüglich Einspoolen, Suchen, Ändern und Wiederzusammensetzen komplexer XML-Dokumente für vier völlig unterschiedliche XML-Datenbanksysteme vor: Tamino, eXcelon, Poet und Oracle, letztere in beiden Varianten: nested Tables und Varrays. Bei der Auswertung der Messungen zeigt sich, dass Oracle8i unter den genannten Vorraussetzungen am besten geeignet ist für die Arbeit mit XML-Dokumenten. Poet liegt im Zeitverhalten dagegen weit dahinter.

Literatur

[1] S. Abitelboul, P. Buneman, D. Suciu: *Data on the Web: from Relations to Semistructured Data and XML,* Morgan Kaufmann, 1999

[2] T. Böhme, E. Rahm: *Xmach-1: A Benchmark for XML Data Managemen,* Datenbanken in Büro, Technik und Wissenschaft (BTW 2001, Oldenburg), Springer, Berlin, 2001, pp. 264-273

[3] R. Bourret: *XML Database Products,* http://www.rpbourret.com/xml/XMLDatabaseProds.htm (April 2001)

[4] A. Schmatloch: *Nutzung von XML zum universellen Datenaustausch zwischen Applikationen,* Diplomarbeit, TU Ilmenau, 2000, InvNr.: 2000-11-08/0007/JN95/2254

[5] A.R. Schmidt, F. Waas, M.L. Kersten, D. Florescu, I. Manolescu, M.J. Carey, R. Busse: *The XML Benchmark Project,* Technical Report INS-R0103, CWI, Amsterdam, April 2001, ISSN 1386-3681